15-213/18-213/15-513: Introduction to Computer Systems

Brian Railing

Carnegie Mellon University
Summer 2016

1 Organization

Class Web page: `http://www.cs.cmu.edu/~213`

Electronic copies of class handouts and lecture slides can be found on the class Web page.

Teaching staff email address: `15-213-staff@cs.cmu.edu`

Please send email to this address whenever you have questions about the course administration, such as extensions in special circumstances or regrading. Don't send mail to individual staff members except to schedule one-on-one meetings. Any emails you send to this address will be received by all members of the teaching staff. Using it allows us to give you the fastest and most consistent responses to your questions.

Questions about the material should be made privately on Piazza.

Instructors:

Brian Railing
bpr@cs.cmu.edu
GHC 6005

Lecture: Tue, Wed, Thu, 12:00–1:20pm, GHC 4215

Note: 15-513 students can view videotaped lectures using the link on the course web page.

Office Hours: Please see the class Web page for instructor and TA office hours.

2 Objectives

Our aim in 15-213 is to help you become a better programmer by teaching you the basic concepts underlying all computer systems. We want you to learn what really happens when your programs run, so that when things go wrong (as they always do) you will have the intellectual tools to solve the problem.

Why do you need to understand computer systems if you do all of your programming in high level languages? In most of computer science, we're pushed to make abstractions and stay within their frameworks. But, any abstraction ignores effects that can become critical. As an analogy, Newtonian mechanics ignores relativistic effects. The Newtonian abstraction is completely appropriate for bodies moving at less than $0.1c$, but higher speeds require working at a greater level of detail.

The following "realities" are some of the major areas where the abstractions you've learned in previous classes break down:

1. *Int's are not integers, Float's are not reals.* Our finite representations of numbers have significant limitations, and because of these limitations we sometimes have to think in terms of bit-level representations.

2. *You've got to know assembly language.* Even if you never write programs in assembly, The behavior of a program cannot be understood sometimes purely based on the abstraction of a high-level language. Further, understanding the effects of bugs requires familiarity with the machine-level model.

3. *Memory matters.* Computer memory is not unbounded. It must be allocated and managed. Memory referencing errors are especially pernicious. An erroneous updating of one object can cause a change in some logically unrelated object. Also, the combination of caching and virtual memory provides the functionality of a uniform unbounded address space, but not the performance.

4. *There is more to performance than asymptotic complexity.* Constant factors also matter. There are systematic ways to evaluate and improve program performance.

5. *Computers do more than execute instructions.* They also need to get data in and out and they interact with other systems over networks.

By the end of the course, you will understand these "realities" in some detail. As a result, you will be prepared to take any of the upper-level systems classes at Carnegie Mellon (both CS and ECE). Even more important, you will have learned skills and knowledge that will help you throughout your career.

3 Textbook

The primary textbook for the course is

> Randal E. Bryant and David R. O'Hallaron, *Computer Systems: A Programmer's Perspective, Third Edition (CS:APP3e)*, Pearson, 2016.

Please make sure you have the Third Edition, which is significantly different from the Second Edition published in 2011. In addition, we require you to have the following reference book on the C programming language:

> Brian W. Kernighan and Dennis M. Ritchie, *The C Programming Language, Second Edition*, Prentice Hall, 1988.

This is the classic *K & R* book, the standard against which all reference manuals are compared. It is an essential part of every computer scientist's library.

4 Course Organization

Your participation in the course will involve these forms of activity:

1. 15-213 and 18-213: Attending the lectures
2. 15-513: Viewing the videotaped lectures
3. Doing laboratory assignments.
4. Reading the text.
5. Taking exams.

Attendance will not be taken at the lectures or recitation sections. You will be considered responsible for all material presented at the lectures and recitations.

Lectures will cover higher-level concepts. Recitations will be more applied, covering important "how-to's", especially in using tools that will help you do the labs. In addition, the recitations will help clarify lecture topics and describe exam coverage.

The textbook contains both *practice problems* within the chapter text and *homework problems* at the end of each chapter. The intention is that you work on the practice problems as you are reading the book. The answers to these problems are at the end of each chapter. Our experience has been that trying out the concepts on simple examples helps make the ideas more concrete. In addition, the class schedule (on the class Web page) shows specific homework problems with each lecture topic. The intention is that you try these out and discuss them in the next recitation. You will find that you will get much more out of recitation if you have done some advance preparation.

The only graded assignments in this class will be a set of seven labs. Some of these are fairly short, requiring just one week, while others are more ambitious, requiring several weeks.

5 Getting Help

For all communication with the teaching staff, please send email to `15-213-staff@cs.cmu.edu`.

This mailing list reaches all of the teaching staff. When you have questions about assignments, please don't send email to individual staff members. Use Piazza instead to maximize your chances of getting a speedy reply.

We will use the class website (`http://www.cs.cmu.edu/~213`) as the central repository for all information about the class.

The lab assignments are offered through a hosted autograding service, developed by Dave O'Hallaron and a group of CMU undergrads, called *Autolab*. See the Autolab Web page at `http://autolabproject.com` for more information.

If you want to talk to a staff member in person, the posted office hours are the best opportunity, as the they represent times when we guarantee that we will be in the location identified. If a meeting is needed outside of the office hours, please use email to arrange a time.

6 Policies

Working Alone on Assignments

You will work on all assignments by yourself.

Handing in Assignments

All assignments are due at 11:59pm (one minute before midnight) on the specified due date. All handins are electronic using the Autolab system. You may handin in as often you like, with your most recent handin counting for credit.

Handing in Late Assignments

The penalty for late assignments is 15% per day. Each student will receive a budget of five *grace days* for the course. These grace days are provided to allow you to cope with most emergencies or other scheduling issues that prevent completing a lab on time, including computer problems, a cold, getting stuck at the airport, etc. You should not plan on using them. Here is how grace days work:

- Grace days are applied automatically until you run out. But, no more than two grace days can be used on any one assignment.
- If your last handin is one day late, and you have at least one remaining grace day, then you will receive full credit for the lab and automatically spend one grace day. For example, if an assignment is due at 11:59pm on Thursday and your last handin is noon on Friday, then you will receive full credit and spend one grace day.
- Once you have spent your grace days, or exhausted the limit for the assignment in question, then you will receive a penalty of 15% for each subsequent late day. For example, suppose you have only one grace day left. If an assignment is due at 11:59pm on Thursday and your last handin is noon

on Saturday, then you will spend your one remaining grace day and be penalized 15%. If your last handin is noon on Sunday, then you will spend one grace day and be penalized 30%.

- Handins will not be accepted after the *end date* of the lab, which is typically three days after the due date.
- The budget of five grace days is for the first six assignments (through malloclab). There are no grace days for the final assignment (proxylab). You can use up to one grace day for the first two assignments (datalab, bomblab), and up to two grace days for the later assignments (attacklab, cachelab, tshlab, malloclab).

Grace days are a tool to allow you to manage your time in the face of personal issues and to help smooth out burstiness in assignment due dates across classes. They are for when you are sick, when a short-term emergency situation arises, when you have too many deadlines all at once, etc. Except for serious persistent personal issues (see below), you should not anticipate additional deadline leniency. **We strongly recommend that you conserve your grace days, saving them for the more difficult assignments at the end of the term.**

Dealing with Serious Persistent Personal Issues

We hope that everyone in 15-213 will remain happy and healthy. But, if you have a serious persistent personal issue, such as being hospitalized for an extended period or needing to leave the country for a family matter, please talk to your academic advisor as soon as possible. Such issues consistently affect one's ability to succeed in all classes, rather than just 15-213, and the academic advisors are equipped to coordinate plans for dealing with them. We will cooperate with such plans, but we cannot construct them independently of the academic advisors.

Requesting a Regrade for an Assignment or an Exam

After each exam and lab assignment is graded, your score will be posted on the Autolab gradebook. We will make the utmost effort to be fair and consistent in our grading. But, we are human. If you believe that you did not receive appropriate credit for an assignment or an exam, you may request a regrade as follows:

- **Exam regrade request:** Submit your hardcopy exam with a cover letter explaining in detail why you think that there was a mistake in the grading.
- **Lab regrade request:** Submit a cover letter (no hardcopy of your code required) explaining in detail why you think there was a mistake in the grading.
- All regrade requests must be submitted to Prof Railing both in hardcopy (GHC 6005) and per email (bpr@cs.cmu.edu). Slide them under his door if he is not in. **Please note that verbal or email requests will NOT be accepted.**
- All regrade requests must be received within **seven days** of the grades becoming available. (Note that the exams provide your grade to you immediately.)

Your request will be processed off-line, and we will respond to your request as quickly as possible (typically within a week). This regrade policy is designed to correct legitimate mistakes in grading, while discouraging frivolous regrade requests (for the sake of being fair and consistent across the entire class).

Final Grade Assignment

Each student will receive a numeric score for the course, based on a weighted average of the following:

- **Assignments (50%):** There are a total of seven assignments (labs), which will count a combined total of 50% of your score. Assignments have different weightings, based on our perception of the relative effort required. See the class Web page for the assignment weightings.
- **Exams (50%):** There will a midterm exam counting 20% and a final exam counting 30%. In summer, 15-513 students have a 10 and 40% split respectively.

Grades for the course will be determined by a method that combines both a small amount of curving and absolute standards. The total score will be plotted as a histogram. Cutoff points start at 90, 80, ..., and are further determined by examining the quality of work by students on the borderlines. Individual cases, especially those near the cutoff points may be adjusted upward based on factors such as attendance, class participation, improvement throughout the course, final exam performance, and special circumstances. If you are above 90.0%, you will receive an A and similarly for each succesive 10%.

Cheating

Please read this carefully, especially if this is your first semester at CMU!

Each exam and lab assignment must be the sole work of the student turning it in. Assignments will be closely monitored by automatic cheat checkers, including comparing turned-in code to the work of students from the same and previous semesters, and students may be asked to explain any suspicious similarities. These cheat checkers are very effective, having been refined over years of research, and they are not fooled by attempts to mask copying of code. Please don't try your luck.

The default penalty for cheating is to be removed from the course with a failing grade. The University gives the instructor some leeway to adjust the penalty. All cases will include the University placing a record of the incident in the student's permanent record.

No collaboration on exams is allowed. The following are guidelines on what non-exam collaboration is authorized and what is not:

What is Cheating?

- *Sharing*: Sharing code, either by copying, retyping, looking at, or supplying a copy of a file from this or a previous semester. Be sure to store your work in protected directories / repositories, and log off when you leave an open cluster, to prevent others from copying your work without your explicit assistance.
- *Describing*: Verbal description of code from one person to another.

- *Coaching*: Helping your friend to write a lab, line by line.
- *Searching*: Searching the Web for solutions.
- *Copying:* Copying code from the Web or another student. You are only allowed to use code that we provide you or that is provided on the CS:APP Web site.
- *Reusing:* Code reuse is complex. You are allowed to reuse general knowledge pieces from prior courses. For example, you can reference your own code for a linked list or how to process commandline arguments. You may not submit your own prior work of the labs, even if they are your own, if you received credit for that course either at CMU or at another institution. If you dropped the course before it was completed, then it is not cheating to reuse your submissions. It is highly discouraged.
- *Looking at other's code:* Although mentioned above, it bears repeating. Looking at other students' code or allowing others to look at yours is cheating. This includes one person looking at code and describing it to another. There is no notion of looking "too much," since no looking is allowed at all.

What is NOT Cheating?

- Clarifying ambiguities or vague points in class handouts or textbooks.
- Helping others use the computer systems, networks, compilers, debuggers, profilers, or other system facilities.
- Helping others with high-level design issues only. Algorithm implementations and other such details are not "high-level design issues". If you need code to describe the issue, then it is not high-level.
- Helping others with high-level (not code-based) debugging.
- Using code from the CS:APP website or from the class Web pages is always OK.

Be sure to store your work in protected directories, and log off when you leave an open cluster, to prevent others from copying your work without your explicit assistance.

7 Mobile devices and other distractions

Research on learning shows that unexpected noises and movement automatically divert and capture people's attention, which means you are affecting everyone's learning experience if your cell phone, pager, laptop, etc. makes noise or is visually distracting during class. For this reason, we allow you to take notes on your laptop, but insist that you turn the sound off so that you do not disrupt other students' learning. If you are doing anything other than taking notes on your laptop, please sit in the back row so that other students are not distracted by your screen.

8 No recording of class meetings

Recordings of any 213 class, in part or whole, including any audio and/or video recordings, regardless of the media or format, and regardless of the intended or actual use, are not permitted without explicit prior written consent of all instructors. The class will be notified in advance should any such recording be approved. Students have no right to record classes under any University policy. If a student believes that he/she is disabled and needs to record or tape classroom activities, he/she should contact the Office of Equal Opportunity Services, Disability Resources to request an appropriate accommodation.

The penalty for violating this policy is an R in the course. If you are not comfortable with this, please drop the course now.

This policy is intended primarily to protect the privacy of the students. For example, no student should run the risk of potential employers finding a question, incorrect answer, or even look of confusion on the Web. The classroom is a learning environment, not an exhibition. Rather than attempt to control the uncontrollable or distinguish between neutral and detrimental uses, all recording is prohibited. Experience has shown that, excluding special cases such as use by students with disabilities or distance learners, undergraduate students do not improve their performance through the use of recordings.

9 Facilities: Intel Computer Systems Cluster

Intel Corp. has generously donated a cluster of Linux-based 64-bit multicore Nehalem servers, specifically for 15-213, that we will use for all labs and assignments. The class Web page has details.

10 Class Schedule

Please see the schedule maintained on the class Web page for information about lectures, reading assignments, suggested homework problems, lab start and end dates, and the lecturer for each class. The reading assignments are all from the CS:APP3e book.